USER'S GUIDE TO ADD NEW GFL VARIABLES OR NEW GFL ATTRIBUTES IN ARPEGE/IFS, ALADIN, AROME: CYCLE 41.

YESSAD K. (METEO-FRANCE/CNRM/GMAP/ALGO)

August 6, 2014

Contents

1 Introduction. **2**

2 Basics about GFL dataflow. **2**

3 Add a new GFL variable: main features. **3**

4 Add a new GFL variable: additional features when read/write GFL on ARPEGE files. **6**

5 Add a new GFL variable: additional features when post-processing of GFL. **7**

6 Add a new GFL variable: additional features when assimilating GFL. **8**

7 Add a new GFL variable: additional features when applying horizontal diffusion to GFL. **9**

8 Add a new GFL variable: additional features when applying nudging to GFL. **9**

9 Add a new GFL variable: additional features for LAM models (coupling and spectral nudging). **9**

10 Add a new GFL attribute. **10**

11 References. **11**

1 Introduction.

The GFL structure has been introduced since several years for "conservative" prognostic variables and also pseudo-historical variables, and more and more GFL variables have been recently introduced. Some recent contributions show that people involved in the introduction of new GFL variables or attributes have not always a comprehensive knowledge of the GFL data flow, the consequence being misplaced pieces of code or forgotten pieces of code able to generate "hidden" bugs. This is due to the fact that some parts of the code deal with GFL individually and not in a global way. Adding GFL is not so easy that one could believe and the purpose of the present paper is to provide a user's guide for people wanting to introduce new GFL variables or new GFL attributes. This is not a comprehensive documentation of all the GFL features. A partial description of the GFL data flow can be found for example in (Tudor, 2004).

GFL variables can be divided into two main classes: the genuine prognostic variables (like specific humidity) and the pseudo-historical variables. Some of them are simply diagnosed in the code, some other ones have an initial state which is read on a file. Some of them can be post-processed. Some of them can be assimilated. Some of them can be diffused (in spectral space). There are individual GFL (for example specific humidity) and groups of GFL (for example aerosols "AERO").

We first start to list the common features to be added when introducing GFL (read or not on a file, post-processed or not, assimilated or not). Some details will be then given to describe:

- how to read/write GFL on files.
- how to post-process GFL.
- how to assimilate GFL.
- how to apply horizontal diffusion on GFL.
- how to apply nudging on GFL.
- how to add new attributes.

2 Basics about GFL dataflow.

Structures are defined in the following modules:

- yom_ygfl.F90 contains:
  - TYPE_GFL_COMP for attributes.
  - TYPE_GFL_NAML: subset of attributes which are in namelist NAMGFL.
  - TYPE_GFLD for dimensions and variables typed by TYPE_GFL_COMP and TYPE_GFL_NAML.
  - maximal number of GFL variables in a sub-group (example JPTRAC).
  - TYPE_GFL_COMP typed variables YGFL%Y[X] for each variable [X], and YGFL%YCOMP.
  - TYPE_GFL_NAML typed variables read in NAMGFL: YGFL%Y[X]_NL for each variable [X].
  - TYPE_GFLD typed variable YGFL.
- type_gflflds.F90: additional type definitions, used currently in FULL-POS.
- yomgfl.F90 contains:
  - buffers, for example GFL, GFLT1.
  - TGFL type definition: allows to gather all variables declared in yomgfl.F90 in one global structure.
  - several encapsulated subroutines (like ZERO_YOMGFL) which do operations on GFL buffers.
- yomgfl5.F90 (trajectory version of YOMGFL) contains:
  - buffers, for example GFL5, GFL_DEPART.
  - TGFL5 type definition: allows to gather all variables declared in yomgfl5.F90 in one global structure.
- gfl_subs_mod.F90 contains:
  - additional declarations (complementary to yom_ygfl.F90 ones).
  - several encapsulated setup routines (to fill attributes of YGFL%Y[X]_NL, YGFL%Y[X], YGFL%YCOMP), allocation and deallocations routines.

All the set-up is now gathered under SU0YOMA − > SUGFL. SUGFL calls SUGFL1, SUGFL2 and SUGFL3.

- SUGFL1 − >
  - default value for some dimensions declared in yom_ygfl.F90.
  - SUDEFO_GFLATTR: default values for YGFL%Y[X]_NL attributes (often depend on LECMWF or the physics package used).
  - reads NAMGFL.
  - SUCTRL_GFLATTR: does checkings (and sometimes resettings) of YGFL%Y[X]_NL attributes.
  - reads NAMCHEM if there are chemistry fields activated (NCHEM>0).
- SUGFL2: partially fills YGFL%Y[X] and YGFL%YCOMP; calls DEFINE_GFL_COMP (encapsulated in gfl_subs_mod.F90) for each GFL variable for that.
- SUGFL3: ends to fill YGFL%Y[X] and YGFL%YCOMP; calls SET_GFL_ATTR (encapsulated in gfl_subs_mod.F90) for each GFL variable for that.

3 Add a new GFL variable: main features.

∗ **List of routines to update:** For the time being we assume that we do not add any new attribute. There is a minimal list of routines which should be updated:

- module/gfl_subs_mod.F90
- module/type_gflflds.F90
- module/yom_ygfl.F90
- namelist/namgfl.nam.h
- setup/sugfl1.F90
- setup/sudefo_gflattr.F90
- setup/suctrl_gflattr.F90
- setup/sugfl2.F90
- setup/sugfl3.F90
- setup/sudyn.F90
- adiab/cptend.F90
- adiab/cptend_new.F90
- adiab/cptendsm.F90
- adiab/cptend_flex.F90
- adiab/gprcp.F90
- phys_dmn/cpchet.F90
- phys_dmn/mf_phys.F90
- adiab/cpg_pt.F90
- module/yomphyder.F90, phys_ec/ec_phys.F90, phys_ec/callpar.F90 and routines under phys_ec/callpar.F90 (for ECMWF applications only).

∗ **module/gfl_subs_mod.F90:** Some GFL are linked to clouds (for the time being, liquid water, ice, cloud fraction, rain, snow, graupels, hail, convective precipitation flux), and in some applications they need to be deallocated (routine DEACT_CLOUD_GFL) or reallocated (routine REACT_CLOUD_GFL). If a new GFL (of generic name [VGFL]) enters this series of cloud variables, some code must be added in module/gfl_subs_mod.F90 at three locations: declaration of Y[VGFL]_SAVE, in routine DEACT_CLOUD_GFL and in routine REACT_CLOUD_GFL.

∗ **module/type_gflflds.F90:** New GFL variables must be added in the type definitions present in this module.

∗ module/yom_ygfl.F90:

- Value of JPNAMED_GFL must be updated: this is the number of individual GFL.
- If an existing group (generic name [GRUP]) has to include some new GFL variables, JP[GRUP] must be updated.
- If a new group (generic name [GRUP]) has to be created, the new variable JP[GRUP] must be created.
- Type TYPE_GFLD:
  - For a new individual GFL (generic name [VGFL]), TYPE_GFL_COMP-typed attribute Y[VGFL] must be added.
  - For a new group of GFL (generic name [GRUP]), TYPE_GFL_COMP-typed attribute Y[GRUP](:) must be added.
  - For a new group of GFL (generic name [GRUP]), variable N[GRUP] (or NGFL_[GRUP]) must be added.
  - Sometimes a new logical variable must be declared.
  - For a new individual GFL (generic name [VGFL]), TYPE_GFL_NAML-typed attribute Y[VGFL]_NL must be added.
  - For a new group of GFL (generic name [GRUP]), TYPE_GFL_NAML-typed attribute Y[GRUP]_NL(JP[GRUP]) must be added.
  - Dimension of YEXT_NL when a new group [GRUP] is added: JP[GRUP] should be subtracted to the original dimension.

∗ namelist/namgfl.nam.h:

- For a new individual GFL (generic name [VGFL]), variable Y[VGFL]_NL must be added.
- For a new group of GFL (generic name [GRUP]), variable Y[GRUP]_NL must be added.

∗ setup/sugfl1.F90:

- Part 1.1: for a new group of GFL (generic name [GRUP]), a default value must be given to the new variable N[GRUP].
- Part 1.2: in some cases this part may have to be modified.
- Parts 5, 6, 7: in some cases these parts may have to be modified.
- Part 8: update the value of NGEMS when relevant.

∗ setup/sudefo_gflattr.F90:

- Part 1a: for a new individual GFL (generic name [VGFL]), a default value must be given to all the attributes of Y[VGFL]_NL.

  Rules of coding are the following ones: for each attribute and each case, ALL the individual GFL appear and ALWAYS in the same order.

  Attribute LT1 must be T for allocated prognostic GFL, F otherwise.
- Part 1b: for a new group of GFL (generic name [GRUP]), a default value must be given to all the attributes of Y[GRUP]_NL, for all values of JGFL in 1 to JP[VGFL]. Attributs SHOULD always appear in the same order, this is the only way to easily check that there is no attribute forgotten or repeated several times. The order currently used is LGP,LSP,NREQIN,REFVALI,LREQOUT,LGPINGP,LTRAJIO, LDIAG,LADV,LADV5,LCDERS,LT9,LT1,LT5,LPT,LPC,CNAME,IGRBCODE,NCOUPLING,REFVALC, LTDIABLIN,LHORTURB,LSLHD,LCOMAD,LVSPLIP,LQM,LQMH,LHV,LINTLIN,LPHY.

∗ setup/suctrl_gflattr.F90:

- Part 1: some consistency checks for GFL attributes:
  - 1.1: Attribute LVSPLIP:
    if the "LVSPLIP" semi-Lagrangian interpolator is not coded for the new GFL, the test generating an ABOR1 must be updated; add the new GFL in the definition of the final value of LVSPLIP.
    if the "LVSPLIP" semi-Lagrangian interpolator is allowed for the new GFL, a test must be added to check that the other attributes linked to semi-Lagrangian interpolators are .F. (see what is coded for ozone).

  - 1.2: Attribute LHV:
    if the "LHV" semi-Lagrangian interpolator is not coded for the new GFL, the test generating an ABOR1 must be updated.
    if the "LHV" semi-Lagrangian interpolator is allowed for the new GFL, a test must be added to check that the attribute LSLHD is .F. (see what is coded for ozone).
  - 1.3: Attribute LSLHD:
    it is a priori assumed that the "LSLHD" semi-Lagrangian interpolator is coded for the new GFL; add the new GFL in the definition of the final value of LLSLHD.
  - 1.4: Attribute LGP:
    It should be .T. for some GFL variables (no spectral representation).
    For some grid-point GFL, we should ensure that the Eulerian horizontal advections are not called (check that if NSTOP>0, the semi-Lagrangian scheme is switched on).
  - 1.5: Attribute LADV:
    For some of the GFL variables, LADV can be set to .T. only if LSLAG=T (because they are grid-point GFL).
  - 1.6: Attribute LCOMAD:
    it is a priori assumed that the "LCOMAD" semi-Lagrangian interpolator is coded for the new GFL; add the new GFL in the definition of the final value of LLCOMAD;
    add the new GFL in the testing saying that SLHD and COMAD interpolations are not both switched on.

- Part 2: some consistency checks for GRIB codes for GFL: some additional consistency checks can be added when required (case .NOT.LARPEGEF).
- Part 3: reset the GFL attributes LSP and LGP for configurations where there is no GFL at all: reset the attributes LGP and LSP for the new GFL introduced.
- Part 4.1: check for all GFL that LGP and LSP are not both TRUE: do the checking for the new GFL introduced.
- Part 4.2: check for all grid-point GFL required in the AROME physics that, when the AROME physics is switched on, the attribute LGP is set to .T. for all these GFL; if a new grid-point GFL required in the AROME physics is added, the test should be added for this new GFL.
- Part 4.3: some GFL should be present when the ALARO prognostic convection scheme is switched on; if a new GFL required in the ALARO prognostic convection scheme is added, the test should be added for this new GFL.
- Part 4.3b: some convective precipitation GFL should be present when option LGPCMT is T; if a new GFL required for option LGPCMT is added, the test must be updated for this new GFL.
- Part 4.4: check that attribute LTDIABLIN is T only for advected GFL; LTDIABLIN=T has a sense only in a semi-Lagrangian job.
- Part 4.5: check that attribute LHORTURB is T only for advected GFL and when L3DTURB=T; LHORTURB=T has a sense only in a semi-Lagrangian job.
- Part 4.6: check that attribute LINTLIN is T only for advected GFL; LINTLIN=T has a sense only in a semi-Lagrangian job.
- Part 4.8: add additional checkings for the new GFL introduced, in particular when the spectral or grid-point should be present for some options (for example when some physics packages are used).
- Part 5: add a new item checking that N[GRUP] is not greater than JP[GRUP] if a new group of GFL is introduced. When relevant, additional tests on N[GRUP] should be coded in this part.

∗ **setup/sugfl2.F90:** There are three parts (1.1, 1.2 and 1.3) which must be updated. Order of the GFL SHOULD be the same in these three parts (but the order can be different of the one of part 1 of SUGFL3).

- For purely grid-point GFL, part 1.3 must be updated by a call to ABOR1 if the attribute LSP=T. This ABOR1 SHOULD be in part 1.3, never in part 1.2.
- For some groups of GFL, it may exist a constraint like "the GFL of the group are either all grid-point ones or all spectral ones": if a new group of GFL has this constraint, a test must be done in part 1.2 to check if, when there is at least one of them which is purely a grid-point one, all of them are grid-point ones. The corresponding test and CALL ABOR1 should be done in part 1.2.

∗ **setup/sugfl3.F90:**

- Part 1: Call to SUCSLINT, calculation of LLPT, LLPC, and call of SET_GFL_ATTR must be coded for the new GFL added.

∗ **setup/sudyn.F90:**

- In part 3.15 the only GFL checking currently remaining is about the moisture convergence. Only the checkings requiring YOMDYN/NAMDYN variables are allowed to be done in SUDYN (and they should be in part 3 of SUDYN). All the checkings which can be done in SUCTRL_GFLATTR should go in SUCTRL_GFLATTR.

∗ **adiab/cptend.F90, adiab/cptend_new.F90, adiab/cptendsm.F90, adiab/cptend_flex.F90:** Introduction of the new GFL if they are true prognostic ones and if they have a diabatic tendency (MF physical packages).

∗ **adiab/gprcp.F90:** There are still calls to GPRCP which require individual GFL variables. These calls to GPRCP may have to be modified for prognostic GFL which enter the definition of moist air constant.

∗ **phys_dmn/cpchet.F90:** cf. adiab/cptend.F90.

∗ **phys_dmn/mf_phys.F90:** The following updates may be required according to the type of physics called.

- call to INIT_APLPAR, APLPAR, APLPAR_AROME, INITAPLPAR, WRITEPHYSIO, PROFILECHET: take account of the new GFL variable if required; compute the diabatic flux for the new GFL variable if required: if relevant, APLPAR, APLPAR_AROME and some of their callees must be updated too.
- call to CPTEND, CPTEND_NEW, CPCHET. take account of the new GFL variable if required (true prognostic GFL with a diabatic tendency).
- in MF_PHYS pseudo-prognostic GFL updated by a physical flux at each time step are currently not treated like true prognostic GFL (compute the tendency inside CPTEND or CPTEND_NEW and update GFLT1 in CPUTQY): GFLT1 is updated in part 2.9 of MF_PHYS. For a new pseudo-prognostic GFL containing a time-dependent physical flux, an update must be added in part 2.9.
- in some cases additional operations may be required to fill arrays ZTENDGFL or ZTENDGFLR.

∗ **module/yomphyder.F90, phys_ec/ec_phys.F90 and phys_ec/callpar.F90 (and probably some routines under callpar.F90):** Introduction of the new GFL if they are true prognostic ones and if they have a diabatic tendency for the ECMWF physics.

∗ **adiab/cpg_pt.F90:** Introduction of the new GFL if they are true prognostic ones, if they have a diabatic tendency at least for one of the physical packages, and if they are supposed to be used with a predictor-corrector scheme (attribute LPT=T in this case): in this routine, currently GFL is treated globally but the diabatic tendencies are treated individually for each GFL. Note that this routine is now used only for the interface with ECMWF physics.

∗ **adiab/cpg_pt_ulp.F90:** Modern version of cpg_pt.F90, used for the interface of physics called under MF_PHYS. This routine uses GFL arrays so there is nothing to do when adding new GFL variables.

∗ **adiab/cpg.F90:** This routine now passes GFL arrays to its callees so there is nothing to do when adding new GFL variables.

∗ **module/yomnsv.F90:** to be modified only if the list of AROME chemicals (currently put in the GFL "EXT") is modified.

4 Add a new GFL variable: additional features when read/write GFL on ARPEGE files.

There is a minimal list of routines which should be updated:

- module/yomfa.F90
- namelist/namfa.nam.h
- setup/sufa.F90

∗ **module/yomfa.F90:**

- Type FAD: for the new individual GFL [VGFL], add variable YFA[VGFL]. Such variables are not used for groups of GFL, where the GFL attribute CNAME is directly used.

∗ **namelist/namfa.nam.h:** For the new individual GFL [VGFL], add variable YFA[VGFL].

∗ **setup/sufa.F90:** For the new individual GFL [VGFL], variable YFA[VGFL] must be setup in part 1, and printed at the end of part 3.

5 Add a new GFL variable: additional features when post-processing of GFL.

This is not compulsory to make the new GFL post-processable, but if desired there is a minimal list of routines which should be updated:

- module/parfpos.F90
- module/yomafn.F90
- namelist/namafn.nam.h
- setup/suafn1.F90
- setup/suafn2.F90
- setup/suafn3.F90
- fullpos/endpos.F90
- fullpos/endpos_prepgfl.F90
- fullpos/fpcordyn.F90
- pp_obs/pos.F90
- pp_obs/pos_prepgfl.F90
- fullpos/vpos_prep.F90
- and in some cases: fullpos/phymfpos.F90.

∗ **module/yomafn.F90 and module/parfpos.F90:**

- Type FULLPOS_TYPE: One finds variables TFP_[VGFL] for some individual GFL variables (for example TFP_SN for snow, TFP_O3MX for ozone). That means that a new TFP_[VGFL] must be added. For the new group of GFL [GRUP] a new variable TFP_[GRUP](JPOS[GRUP]) must be added.
  Parameter variable JPOS[GRUP] must be added in parfpos.F90.
  Variable JPOS3DF should be updated in parfpos.F90.
  When there are new post-processable fields in an already existing group, JPOS[GRUP] should be updated in parfpos.F90.

∗ **namelist/namafn.nam.h:** One finds variables TFP_[VGFL] for some individual GFL variables (for example TFP_SN for snow, TFP_O3MX for ozone). That means that a new TFP_[VGFL] must be added.

∗ **setup/suafn1.F90, suafn2.F90, suafn3.F90:**

- In SUAFN1, default value for the new TFP_[VGFL] or TFP_[GRUP] (topic GFL, part 0 bis) must be added, calculation of the attribute ILMOD of the new TFP... (end of SUAFN1) must be added.
- In SUAFN2, routine CTRL_TFP must be called for the new GFL variables (part 1).
- In SUAFN3, routine PRINT_TFP must be called for the new GFL variables (part 1).
- The ordering of GFL (and if possible of all the post-processable variables) must be the same in these three routines.

∗ **fullpos/endpos.F90:**

- Part 1.1.1: for new individual GFL [VGFL], add line:

```
CALL SUPTRPPGFL_ENDPOS(1,IPPGFL,YLPPIC%I[VGFL])
```

∗ **fullpos/endpos_prepgfl.F90:**

- Part 1: define LD_GFL(YDPPIC%I[VGFL]) for new individual GFL [VGFL].
- Part 2: define LDS1_GFL(YDPPIC%I[VGFL]) for new individual GFL [VGFL].
- Part 3: define LDS2_GFL(YDPPIC%I[VGFL]) for new individual GFL [VGFL].
- Part 4: define KGFLCOD(YDPPIC%I[VGFL]) for new individual GFL [VGFL].
- Part 5: define KDIN_GFL(YDPPIC%I[VGFL]) for new individual GFL [VGFL].
- Part 6: define LDIN_GFL(YDPPIC%I[VGFL]) for new individual GFL [VGFL].

∗ **fullpos/fpcordyn.F90:** In the JFLD loop, add an adequate piece of code, for a new individual GFL [VGFL] if it should be corrected (for example if it should be bounded by a lower threshold or an upper threshold), and provide the proper value of ZMIN, ZMAX, ICOR for input of FPHOR12. The same thing must probably be done for groups of GFL but no example is currently provided in FPCORDYN to know how to code it.

∗ **pp_obs/pos.F90:**

- Part 1.1.1: for new individual GFL [VGFL], add line:

```
CALL SUPTRPPGFL_POS(1,IPPGFL,YLPPIC%I[VGFL])
```

- Additional pieces of code may be required for GFL in part 1.2.2, for example if horizontal derivatives are needed.

∗ **pp_obs/pos_prepgfl.F90:**

- Part 1: define LD_GFL(YDPPIC%I[VGFL]) for new individual GFL [VGFL].
- Part 2: define LDAP_GFL(YDPPIC%I[VGFL]) for new individual GFL [VGFL].
- Part 3: define KGFLCOD(YDPPIC%I[VGFL]) for new individual GFL [VGFL].
- Part 4: define KDIN_GFL(YDPPIC%I[VGFL]) for new individual GFL [VGFL].
- Part 5: define LDIN_GFL(YDPPIC%I[VGFL]) for new individual GFL [VGFL].

∗ **fullpos/vpos_prep.F90:**

- Part 1: define YDIN%I[VGFL] for new individual GFL [VGFL], and also derivatives YDIN%I[VGFL]L and YDIN%I[VGFL]M if relevant.
- Part 2: define YDGFL%LL[VGFL] for new individual GFL [VGFL], and also derivatives YDGFL%LL[VGFL]L and YDGFL%LL[VGFL]M if relevant.

∗ **Additional modifications:** Additional modifications may be required, for example if new GFL variables become new input dummy arguments to some GP... routines (example of routines to check: PHYMFPOS).

6 Add a new GFL variable: additional features when assimilating GFL.

There are several aspects to be considered, at least the GOMS dataflow and the observation interpolator.

∗ **List of routines to update in the GOMS dataflow:** There is a minimal list of routines which should be updated:

- module/gom_mod.F90 (references to individual GFL variables appear in several type definitions).
- obs_preproc/sugoms.F90
- a subset of the routines using the following variables of module/gom_mod.F90: GID%[gfl] (pointer for array GOM) must be checked, and in particular those where individual GFL variables are mentioned (mostly in directories op_obs and pp_obs).

∗ **List of routines to update in the observation horizontal interpolator:** Only cobs.F90 and cobsad.F90 may have to be updated (references to individual GFL variables).

∗ **Updating TL and AD codes:** Work done on the direct code must be also done for TL and AD codes.

7 Add a new GFL variable: additional features when applying horizontal diffusion to GFL.

There is currently no GFL attribute to know if a GFL variable is diffused or not (spectral diffusion). The choice of diffusing or not a GFL variable is hard coded and can be done only on spectral GFL. For the time being specific humidity and ozone are diffused if the attribute LSP is TRUE (there are specific keys in NAMDYN to remove diffusion on these variables, see documentation (IDDH)), and the other GFL variables are NEVER diffused (even if spectral).

If a new diffusible spectral GFL is introduced, the following routines must be updated:

- module/yomdyn.F90
- namelist/namdyn.nam.h
- setup/sudyn.F90
- setup/suhdir.F90
- setup/suhdf.F90, setup/suhdf_ec.F90 and aladin/setup/suehdf.F90

8 Add a new GFL variable: additional features when applying nudging to GFL.

Nudging is used for climatic simulations.

There is currently no GFL attribute to know if a GFL variable can be nudged or not. The choice of nudging or not a GFL variable is hard coded and can be done only on humidity. For the time being only specific humidity is nudged in spectral space if the attribute LSP is TRUE, and there is some code to nudge humidity in grid-point space too. Other GFL variables are NEVER nudged.

If a new nudgeable spectral GFL is introduced (with nudging in spectral space), the following routines must be updated:

- module/yomnud.F90
- namelist/namnud.nam.h
- setup/sunud.F90
- adiab/spchor.F90

If grid-point nudging must be applied to a new GFL, one should look at all routines using variables LNUDQG and XNUDQG.

9 Add a new GFL variable: additional features for LAM models (coupling and spectral nudging).

∗ **Coupling:**

Coupling is controlled by attribute NCOUPLING. Coupling is automatically updated for new prognostic GFL variables (simply specify the right default for attribute NCOUPLING); no modification is expected in pieces of code called in coupling.

∗ Spectral nudging:

There is currently no GFL attribute to know if a GFL variable can be nudged or not. The choice of nudging or not a GFL variable is hard coded and can be done only on humidity. For the time being only specific humidity is nudged in spectral space if the attribute LSP is TRUE. Other GFL variables are NEVER nudged.

If a new nudgeable spectral GFL is introduced (with nudging in spectral space), the following routines must be updated:

- module/elbc0a_mod.F90 (add variable SPNUD[X]).
- ald_inc/namelist/nemelbc0a.nam.h (add variable SPNUD[X]).
- coupling/external/espcpl.F90 (add nudging on new GFL variable).
- Side effects are expected on coupling/external/espsc2r.F90 and aladin/control/espcm.F90.

Later, adding a new attribute SPNUD (working like NCOUPLING) may be studied (no nudging if 0, nudging if above 0); externalisable part of spectral nudging must be rewritten in order to do spectral nudging on a subset of spectral GFL fields, without explicitly naming them.

10 Add a new GFL attribute.

∗ List of routines to update: There is a minimal list of routines which should be updated:

- module/gfl_subs_mod.F90
- module/yom_ygfl.F90
- setup/sudefo_gflattr.F90
- setup/suctrl_gflattr.F90
- setup/sugfl2.F90
- setup/sugfl3.F90
- setup/sudyn.F90
- setup/sucslint.F90

∗ Add a new attribute in TYPE_GFL_NAML:

- This attribute must be added in the definition of TYPE_GFL_NAML in module/yom_ygfl.F90.
- If the attribute is linked to a semi-Lagrangian interpolator (for example add a new interpolator), routine SUCSLINT must be modified and must take account of this new attribute to compute the variable CLSLINT of SUGFL3; update the calls of CLSLINT in SUGFL3.
- This attribute must be added preferably in routine DEFINE_GFL_COMP (input dummy argument and internal code to update YDGFLC, YGFL and YPTRC), or/and in routine SET_GFL_ATTR (input dummy argument and internal code to update YDGFLC and YGFL) if necessary. If the dummy arguments of DEFINE_GFL_COMP are modified, SUGFL2 must be updated. If the dummy arguments of SET_GFL_ATTR are modified, SUGFL3 must be updated.
- This attribute SHOULD get a default value in parts 1a and 1b of SUDEFO_GFLATTR for all GFL variables.
- Additional checkings can be added if required in SUCTRL_GFLATTR after reading NAMGFL.
- Additional checkings can be added if required in part 3 of SUDYN after reading NAMDYN if these checkings use NAMDYN variables (but in this case it would be desirable to move the involved NAMDYN/YOMDYN variables into NAMDYNA/YOMDYNA and to do the checkings in SUCTRL_GFLATTR).

∗ Add a new attribute in TYPE_GFL_COMP:

- This attribute must be added in the definition of TYPE_GFL_COMP in module/yom_ygfl.F90.
- This attribute must be added preferably in routine DEFINE_GFL_COMP (input dummy argument and internal code to update YDGFLC, YGFL and YPTRC), or/and in routine SET_GFL_ATTR (input dummy argument and internal code to update YDGFLC and YGFL) if necessary. If the dummy arguments of DEFINE_GFL_COMP are modified, SUGFL2 must be updated. If the dummy arguments of SET_GFL_ATTR are modified, SUGFL3 must be updated.
- The new attribute must be printed in PRINT_GFL (module/gfl_subs_mod.F90).

- The new attribute must be added in FALSIFY_GFLC (module/gfl_subs_mod.F90).
- If the new attribute should be reset to .F. when there is no advection, it must be added in NOADVECT_GFLC (module/gfl_subs_mod.F90).
- The new attribute must be added in COPY_GFL_COMP (module/gfl_subs_mod.F90).

∗ **Add a new attribute in TYPE_GFLD:**

- This attribute must be added in the definition of TYPE_GFLD in module/yom_ygfl.F90.
- Calculation of this new attribute must be added in DEFINE_GFL_COMP (module/gfl_subs_mod.F90).
- In some specific cases, the value of this new attribute may be modified in some other routines of module/gfl_subs_mod.F90 (SET_GFL_ATTR, PRINT_GFL if the value of this attribute should be printed, DEACT_CLOUD_GFL and REACT_CLOUD_GFL if the value of this attribute should be modified when deallocating/reallocating cloud GFL variables).

∗ **Remarks:** Future rewriting is expected in order:

- to avoid redundant attributes and to allow simplification of some set-up routines.
- to put in NAMGFL only attributes which have to be modified via namelist (normally attributes like LT9, LT1 must not be in NAMGFL).
- to allow a better separation between attributes which must pass in DEFINE_GFL_COMP and those which must pass in SET_GFL_ATTR.

11 References.

- (IDGOMU) Geer, A., 2013:
  GOM user guide: interpolation to observation space. Internal note, 3pp.
- (IDGFL) Tudor, M., 2004: A short description of physics/dynamics interface in the new data flow. Internal note, 21pp.
- (IDDH) Yessad, K., 2014: Horizontal diffusion in the cycle 41 of ARPEGE/IFS. Internal note, available on “http://www.cnrm.meteo.fr/gmapdoc/”.